基于二进制序列的属性访问控制策略检索方法 *

黄恒杰 [1]，潘瑞杰 [2]，王高才 [2†]，秦利忠 [1]

(1. 玉林师范学院 教育技术中心，广西 玉林 537000; 2. 广西大学 计算机与电子信息学院，南宁 530004)

摘　要：在基于属性访问控制中策略中，如何快速响应检索的访问控制请求至关重要，而通过遍历策略集合每条规则中的所有的属性值去匹配相应规则的检索方法是低效的。因此，论文提出一种基于二进制序列的属性访问控制策略检索方法。采用二进制标识和二进制编码表示基于属性的访问控制策略和访问控制请求。通过对二进制标识的逻辑运算选择合适的分组，在组内，通过访问控制请求的二进制编码和所有规则的二进制编码的匹配来查找合适的规则，减少策略集合内规则的属性与访问控制请求属性匹配的过程，从而提高策略检索效率。论文在实验中从策略预处理、策略评估时间和策略检索总时间三个方面类比相似检索方法的效率，实验结果表明，论文提出的策略检索方法具有更高的检索效率。

关键词：云计算；属性访问控制策略；检索效率；匹配；二进制序列

中图分类号：TP393.08　　doi: 10.19734/j.issn.1001-3695.2022.03.0126

Retrieval method of attribute access control policy based on binary sequence

Huang Hengjie[1], Pan Ruijie[2], Wang Gaocai[2†], Qin Lizhong[1]

(1. *Educational Technology Center*, *Yulin Normal University*, *Yulin* 537000, *China*; 2. *School of Computer & Electronic Information*, *Guangxi University*, *Nanning* 530004, *China*)

Abstract: In Attribute-Based Access Control (ABAC) , how to respond quickly to the retrieved access control request is very important, and it is undoubtedly time-consuming to traverse all attribute values in each rule of the policy set until finding the appropriate rule. Therefore, the paper proposes an ABAC retrieval method based on binary sequence, and uses binary identification and binary coding to represent attribute based access control policies and access control requests. Through the logical operation of binary identification, select the appropriate group. In the group, find the appropriate rules by matching the binary code of access control request with the binary code of all rules, reduce the process of matching the attributes of rules in the policy set with the attributes of access control request, and improve the efficiency of policy retrieval. In the experiment, this paper compares the efficiency of similar retrieval methods from three aspects: strategy preprocessing, strategy evaluation time and total strategy retrieval time. The results show that the strategy retrieval method proposed in this paper has higher retrieval efficiency.

Key words: cloud computing; attribute access control policy; retrieval efficiency; matching; binary sequence

0　引言

云计算技术的兴起和发展为现代社会提供了便捷的数据共享和融合计算等服务，也使计算和存储资源得到充分利用。其资源中包含了大量的用户隐私数据，然而相关机构对这些数据的保护却并不令人满意。隐私数据一旦泄露，可能给个人和机构带来不可估量的损失[1]。例如，西太平洋银行(Westpac)的近 10 万客户的私人信息遭到泄露和 Instagram 用户的 4900 万数据遭到曝光，都是源于非法用户的访问和合法用户对资源的越权访问。因此，访问控制技术作为保护资源的有效手段备受国内外研究者们的关注。在访问控制中，通过验证访问者的身份信息和制定合适策略来规范和限制访问请求者的权限，从而保护客体资源。这些策略确保合法用户在复杂的网络环境中访问和使用资源服务，同时阻止非法用户窃取资源和合法用户的非法访问行为等。其中，基于属性的访问控制(attribute-based access control, ABAC)因具有细粒度和灵活性，并能解决大规模的用户激增问题而在云计算场景得到广泛应用。虽然，现有的 ABAC 模型[2]在安全策略的表达和制定上具有较大优势，但如应用到大规模访问策略的云计算环境下还是会存在检索效率低下的问题。根据 ABAC 和 XACML(extensible access control markup language)的相关理论[3]，当用户发起访问客体资源的请求时，策略决策点会调用相关属性进行解析并匹配到所有的策略，这意味着基于 XACML 的访问控制策略的评估需要遍历所有的策略进行判断，其中属性匹配需要对访问控制请求的所有属性信息和策略集合中的规则的属性信息进行比较。其中，信息匹配包括字符串匹配和数值比较，且只有该策略全部的属性信息都和属性访问请求(attribute access request, AAR)匹配才会授予主体访问客体的权限。此外，在对策略集合中的规则逐个匹配的过程中，若每次都是最后一个属性被匹配时，才发现它与 AAR 不一致，如果此类规则太多就会增加策略检索的时间。本文针对上述问题提出一种基于二进制序列的属性访问控制策略检索方法，以提高策略检索效率。

本文其余部分的组织如下：第 1 节相关理论及研究工作介绍，第 2 节对基于二进制序列的属性访问控制策略检索方法进行描述，第 3 节是实验结果与分析，第 4 节是结束语。

收稿日期：2022-03-06；**修回日期：**2022-05-16　**基金项目：**移动边缘计算中具有能耗优化的数据迁移策略研究；国家自然科学基金资助项目(62062007)；玉林师范学院科研项目(2012YJQN29)

作者简介：黄恒杰(1980-)，男，副教授，硕士，主要研究方向为网络能耗优化和网络空间安全；潘瑞杰(1997-)，女，硕士研究生，主要研究方向为网络安全技术；王高才(1976-)，男(通信作者)，教授，博导，硕导，博士，主要研究方向为计算机网络、性能评估和网络安全(wanggcgx@163.com)；秦利忠(1983-)，男，副教授，硕士，主要研究方向为形式化方法.

1 相关理论及研究工作

传统的访问控制由于其本身的一些局限已经不再适合云计算多域的特点。ABAC 根据用户的主体属性、客体属性、环境属性等信息来授予主体访问权限，因其具有动态性、细粒度和灵活性而备受关注。ABAC 可用四元组(S,O,E,P)来表示，其中，S 为主体(subject)、O 为客体(object)、E 为环境(environment)、P 为权限(privilege)。

访问控制策略由主体、客体、环境和权限组成，可形式化定义为：(permit, deny)←(Attr(S),Attr(O),Attr(E),Attr(P))，分别表示相应的属性取值范围。一条访问控制策略通常是由一个或多个访问控制规则组成，表示主体 S 在环境条件 E 下是否被允许对客体 O 进行 P 权限的操作，其具体形式如下所示。

Rule: access(S,O,E,P) ←*f*(SA,OA,EA,PA)

这里，*f* 是一个布尔函数，其作用是根据访问控制规则对具体的主体、客体、环境和权限的属性作出判断。若返回结果为真，则允许访问；否则，拒绝访问。在 ABAC 的授权步骤中，策略判定点(policy decision point, PDP)、策略执行点(policy enforcement point, PEP)、策略管理点(policy administrate point, PAP)和策略信息点(policy information point, PIP)是其最为重要的支撑，整个过程分为两个阶段。

(1)准备阶段：

①首先，PIP 对所有的主体、客体、权限和环境的属性信息进行搜集和管理，并把属性、权限关系进行关联。

②然后，PAP 从 PIP 获得构建 ABAC 策略所需的属性信息和属性-权限关系构建 ABAC 策略。

(2)执行阶段

①当 PEP 接收到主体发送的原始访问请求(Natural Access Request, NAR)时，PEP 向 PIP 请求获取构建 AAR 所需的主体属性、客体属性、环境属性和权限属性来构建基于属性的访问控制请求。

②PDP 根据 PIP 提供的主体属性、客体属性、环境属性信息和权限属性等属性信息对主体的身份信息进行验证。

③根据 PAP 提供的策略查询结果对 PEP 转发来的 AAR 进行判定，决定是否对 AAR 进行授权。

④PAP 将判定结果传递给 PAP，PEP 执行决策结果。

ABAC 的成功部署需要两个关键部分：一是定义良好的策略描述语言；二是为 ABAC 制定一种有效的策略检索方法。其中，前者有利于防止云平台资源数据泄露和非法授权访问，而后者能确保访问控制请求到来时得到及时响应。

一方面，在如何定义良好的访问控制的授权策略描述语言方面，学者们对不同的访问控制模型和应用场景进行研究，提出了 XACML、应用于 Web 服务的基于图的策略可视化描述语言[4]，适用于 RESTful 接口的策略描述语言 RestPL[5]等。其中，较为常用且适用于 ABAC 模型的是 XACML。但在实际的云部署中，各个企业更倾向于自己的策略描述语言。另一方面，对于一些大型的组织，随着 ABAC 的规模越来越大，其用户在同一时刻发出访问请求的数量会也越来越多。因此，有效的检索 ABAC 策略对实时响应用户的 AAR 是至关重要的，也直接影响着用户的体验。

在如何有效地检索 ABAC 策略方面，目前研究人员已经取得了一些成果。这些研究工作大致分为两类：一类是对策略访问控制语言为 XACML 标准的策略检索方法的改进，另一类是根据下一代访问控制 NGAC[6,7]标准进行策略检索方法的改进。

对于如何提高策略的检索效率，缩短策略检索时间，这方面的研究较少。在传统的策略检索方法的检索过程中，当有访问控制请求时，需要对所有的策略进行遍历，若出现策略冗余和冲突，XACML 需支持四种组合算法：(1)优先适用，(2)只适用一项，(3)允许-覆盖，(4)拒绝-覆盖 [8]。为了提高策略的检索效率，文献[9]提出了一种基于图的 XACML 策略评估方法，该方法对匹配树和合并树两个策略评估策略数据结构进行优化，并且针对策略匹配树提出一种二进制查找算法，该方法不支持多值属性。文献[10]提出了一种新的基于统计学分析的策略评估引擎 HPEnigne 来提高分布式环境下 XACML 策略评估引擎的效率，该引擎将策略由文本形式转换成数值形式，缩减了策略规模，同时，采用多缓存机制来提高策略的查找速度。文献[11]提出一种基于前缀的标记运算的策略检索方法，该方法根据策略属性的取值添加二进制前缀，从而缩小策略检索的范围。这种检索方法确实提高了检索效率，但是，在访问控制策略与访问请求属性名匹配的情况下，其前缀计算反而造成时间上的浪费。针对该问题，文献[12]在基于属性访问控制策略添加二进制标识之后，然后在属性值级别引入策略决策树来提高检索效率，并且该方法具有良好的可扩展性，但是该方法在访问控制策略属性比较多的情况存在一定的局限性。文献[13]在[11]的基础上进行研究，提出了一种基于属性分组的访问控制策略检索方法，通过对策略集进行基于属性的分组而缩减策略检索的范围，从而减少不必要的计算以及比较过程，由此实现对访问控制策略的高效率检索。

在下一代访问控制标准中，策略决策点负责在访问控制策略中检索符合访问控制请求的规则[14]。在进行策略检索时，需要把 AAR 的属性和访问控制策略中的每一条规则的属性信息进行比较，直到找到符合的规则，这无疑是费时的。为此，文献[15]提出了一种名为 PolTree 的数据结构，它通过两个变体 PolTree 和 N-PolTree 来存储 ABAC 策略，并减少策略检索时的属性-值对的比较次数，从而提高策略的检索效率。

总的来说，各种 ABAC 检索方法都有优势和缺点，其检索效率与传统的策略检索方法相比，其效率也有所提高，但仍然存在一定的局限性。基于路由中二叉线索树能快速在转发表中找到匹配的叶节点，从而提高查找效率的思想，本文提出一种基于二进制序列的属性访问控制策略检索方法，该方法通过对策略进行二进制标识，根据二进制标识对其进行分组形成若干策略集合组，对组内的策略进行二进制编码。当进行访问时，先对访问控制请求进行二进制标识与编码，然后通过访问控制请求与策略的逻辑运算从而选择合适的组，过滤掉大量无关策略。最后，访问控制请求的二进制编码的值与组内策略的值进行逻辑运算而避免对策略中属性信息的单个匹配，从而提高检索效率。

2 基于二进制序列的属性访问控制策略检索方法

2.1 构建基于二进制标识的策略和 AAR

为了对 ABAC 策略和 AAR 进行二进制标识，本文需要统计出现在访问控制策略集合中所有属性，并对该属性按照主体、客体、环境和权限的特定顺序排列。并且，主体、客体、环境和权限属性的内部属性也要按一定的顺序进行排列，以此来保证二进制标识的有效性。通常，属性分为目录属性和非目录属性，目录属性指的是一些决策信息属性值，包括{allow, denny}，每条规则有且仅有一个决策属性。非目录属性指的是作出决策需要衡量的属性信息包括主体属性、客体属性、环境属性和操作属性。对规则的编码只是针对非目录属性，分别用二进制数字 0 和 1 来表示该属性在访问控制请求和策略集合中是否出现，0 表示未出现，1 表示出现，从而形成 ABAC 策略和 AAR 的二进制标识。

在云计算环境中，用户通过各自的终端设备请求服务，如果本文构建如表 1 所示的属性信息，再利用这些属性信息

去构建如表 2 所示的 ABAC 策略。二进制标识只是针对非目录属性，其维度就是非目录属性的个数，即统计访问控制规则中所有属性个数，其属性的总数就是二进制标识的位数。例如，表 2 中单个规则中拥有的最多的非目录属性个数是 6，所以二进制标识位的长度为 6。规则中存在该条属性，就把对应的值置为 1，否则为 0。也就是说，每一个二进制位表示该条规则中是否存在该属性。

表 1 属性组合

Tab. 1 Attribute combination

主体属性		客体属性		环境属性	权限属性
SA_Role	SA_trust	OA_type	OA_trust	EA_Network	PA_permission
student	low	personal	low	home	delete
teacher	middle	common	middle	work	read
admin	high		high	public	write

表 2 访问控制策略集合

Tab. 2 Access control policy set

规则	attr(SA)	Attr(OA)	Attr(EA)	Attr(PA)	Decide	二进制标识
R_1	student ∨ low	personal ∨low	home	delete	permit	111111
R_2	low	personal ∨ low	public	delete	deny	011111
R_3	Student	low	work	delete	deny	100111
R_4	Student	low	home	delete	permit	100111

如前所述，ABAC 的访问控制分为准备阶段和执行阶段。在准备阶段，PAP 需要对策略集合中的每条规则进行二进制标识。 在策略执行阶段，只需要对 AAR 进行二进制标识，然后与策略的二进制标识进行逻辑与运算，当运算结果与访问控制规则的二进制标识一致时，则对该规则的其他属性进行检测，即判断 AAR 的属性是否符合规则的属性信息。

为进一步减少策略检索时间，根据 ABAC 策略的二进制标识进行分组，构建类似路由中二叉线索树。在该树中，除了第一层根节点外的每个节点都表示一个分组，二进制标识相同的规则在同一组，二进制标识不同的在不同的组。

访问控制规则的属性代表的是对 AAR 的要求，即要跟该条访问控制规则匹配成功，AAR 需要拥有规则所要求的属性和属性值。这就意味着 AAR 和规则的匹配并不非毫无差别的匹配。实际上，在策略检索的时候，只检测 AAR 的属性是否符合访问控制规则的属性信息，对于 AAR 中多出的属性，并不做额外的匹配要求，即 AAR 中的额外属性并不影响其匹配的成功与否。例如，某 AAR 为(SA_Role=”student”, SA_trust=”low”, OA_type=”personal”, OA_trust=”low”, EA_Network=”public”, PA_permisssion=”Denny”)，其二进制标识为 111111，但是，对其二进制标识位 011111，101111，…，111110 的分组也有可能存在符合该 AAR 的规则。若在表 2 的访问控制策略集合中查找合适的规则，其中，二进制标识为 011111 的分组中的 R_2 规则也符合该 AAR。

2.2 属性信息的二进制编码

实际上，如果对 ABAC 策略进行二进制标识并进行分组，在进行策略检索时，就能有效的缩短策略检索的范围，使得策略的范围局限到符合二进制标识的几个组内，过滤掉大量无关的规则，从而缩短策略检索的时间。但是，当在本组检索的时候，需要对每条规则的每个属性的取值与 AAR 的对应取值进行匹配，检验 AAR 是否满足该条规则，最坏情况就是每次检测到规则的最后一个属性时，才发现该规则不符，这样将会造成时间上的浪费。如果存在这样的规则过多，将会造成策略检索时间过长。为此，本文提出对每个组的访问控制策略进行二进制编码。这样只需要对 AAR 的二进制编码与 ABAC 策略的二进制编码进行逻辑运算而作出授权决策，而不需要对策略集合的每条访问控制策略的每个属性值进行判断，从而节省策略检索的时间，提高策略检索效率。

ABAC 的属性值由两种类型组成：离散属性值和连续属性值。

1)离散属性编码

例如，上文中出现的主体属性的 SA_Role 的属性值的取值范围为{student，teacher，admin}是离散属性值。如果主体 s_1 的属性 sa_1 是离散属性，说明 sa_1 的取值是相互独立的，其取值范围可以通过枚举的方式表示。本文对其采用哑变量编码方式进行编码。其中，1 表示该属性值存在，0 表示该属性值不存在。对于该属性位都不存在的情况，则把所有位置全置为 0。假设 sa_1 的取值范围为{sa_1v_0, sa_1v_1, sa_1v_2},一共有 3 个值可取,则采用三个二进制位进行编码。若 sa_1 的取值范围为 attr(sa_1)≠{sa_1v_1},对其编码时除了第二位其余位全置为 0，即编码为 101。若为 attr(sa_1)={sa_1v_1, sa_1v_2},则把二进制序列的第二位和第三位置为 1,其余位置全置为 0，即 110。主体 s_1 的属性 sa_2 的取值范围为{sa_2v_0, sa_2v_1},一共有 2 个值可取。因此,采用 5 个二进制位对主体 s_i 属性进行编码，如表 3 所示。

表 3 属性的二进制编码

Tab. 3 Binary encoding of attributes

主体属性取值范围	二进制编码
attr(sa_1)={sa_1=sa_1v_1}∨attr(sa_2)={sa_2=sa_2v_0}	01010
attr(sa_1)={sa_1≠sa_1v_1}∨attr(sa_2)={sa_2=sa_2v_0}	01101
attr(sa_1)={sa_1={sa_1v_1, sa_1v_2 }∨attr(sa_2)={sa_2=sa_2v_0}	01110
attr(sa_1) ={sa_1=sa_1v_1}	00010
attr(sa2)={sa_2=sa_2v_0}	01000

2)连续属性编码

在特定访问控制环境中会指定在某个时间段内允许访问资源或者需要设置主体属性的信任等级处于每个信任值之间，这就需要设置连续属性。对连续的无法枚举的属性值，需要将其离散化。在访问控制系统中比较常用的连续属性是信任连续属性和时间连续属性。以信任连续属性为例对连续属性如何离散进行说明，时间连续属性和信任连续属性的编码和离散化过程类似。如前述主体的 SA_trust 属性和客体的 OA_trust 属性表示的都是主体所具备的信任值和访问客体的主体需要满足的信任值，主体所具备的信任值是采用一定的计算方法计算出来的值如文献[16]所提出的计算方法。

2.3 基于二进制序列的策略检索方法分析

访问控制策略中每个属性的取值范围代表的是对访问控制请求的属性组合的要求，即想要获得访问客体的特定权限需要满足的属性和属性值，这意味着策略和访问控制请求的匹配并非是完全一致无差别的匹配。实际上，策略的匹配都是字符串的匹配，访问控制策略只检测访问控制请求是否拥有相关属性信息，访问控制请求有额外的属性信息并不影响其匹配的成功与否。例如，存在这样的一条策略，主体可以在工作时间访问客体资源，如果主体在工作时间发出访问客体资源的请求，在该请求中包含了该主体的角色信息，那么该主体的访问仍然是被允许的。

以表 2 中的策略集合为例,选取表 1 中的属性值构建基于属性的访问控制请求{SA_role=student, SA _trust=low, OA_trust=low, EA_Network=work, PA_permission=delete},首先，根据本文提到的基于二进制序列属性访问控制策略检索方法,对表 2 中的规则进行编码的结果依次为：(R_1, 10010010100100100)，(R_2, 00010010100001100)，(R_3, 10000000100010100),(R_4, 10000000100100100)。然后对 AAR 进行二进制标识,除了 OA_type 未设置属性值外,每个属性都有属性值，所以 AAR 的二进制标识为 110111。按照前述提到的编码方法对 AAR 进行编码为(AAR, 10010000100010100)。访问控制请求的二进制标识和策略的二进制标识做逻辑与运算 110111 & 111111=110111≠111111,

说明 R_1 规则所在的组不符合访问控制请求的属性要求，跳过 R_1 所在的组，对第二个组的规则进行查询 110111 & 011111=010111≠011111，所以 R_2 所在的第二组也不符合属性要求，过滤掉该组。对第三个组进二进制标识为 100111 的规则组进行查询，110111 & 100111=100111,说明在 R_3 规则所在的组是需要进一步查找的组，R_3 规则的二进制编码 10000000100010100 与访问控制请求的二进制编码 10010000100010100 进行逻辑与运算的结果为 10000000100010100 策略的二进制编码相等，即 10000000100010100 & 10010000100010100= 10000000100010100，则该条规则就是要找的规则，因为其决策属性为 Denny，所以 PDP 的决策结果将是拒绝本次访问，本次策略检索结束。

在以上的策略检索过程中，如果采用传统的策略检索方法，则需要对策略集合中的每条规则的每个属性进行匹配，直到找到所需要的规则为止，因此需要执行的匹配次数为 18 次，如果采用根据二进制标识分组的方法则需要 AAR 的二进制标识和访问控制策略的二进制标识做逻辑与运算，并判断是否与策略的二进制标识一致，减少了不合适规则属性的匹配次数，因此进行匹配的次数为 10 次。采用基于二进制序列的属性访问控制策略检索方法只需要进行二进制标识匹配和二进制编码匹配，因此只执行的匹配次数为 4 次，其具体的策略检索过程如图 1 所示。

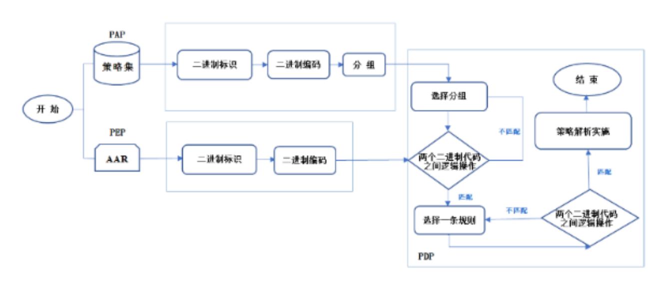


图 1　基于二进制序列的策略检索机制流程

Fig. 1　Process of policy retrieval mechanism based on binary sequence

基于二进制序列的属性访问控制策略检索方法的核心算法如下算法 1。

算法 1　基于二进制标识的策略检索算法

```
Input: policy_set, AAR     /*策略集合，要进行判别的访问控制请求*/
Output: Decision           /*决策结果*/
1. Begin
2. Step 1: Identify and encode attribute-based access control requests.
3. Init(AAR.policy_groupbinary);
4. Init(AAR.policy_code);  /*对访问控制请求二进制标识和策略编码位进行初始化*/
5. for (i=0;i< AAR.size();i++) do
6. if (AAR[i]) exist then
7. AAR.policy_groupbinary.setposition(p)=1 /*对访问控制请求进行二进制标识*/
8. end if
9. end for
10. for (j=0;j<AAR.policy_code.size();j++) do   /*对访问控制请求进行二进制编码*/
11. if (AAR[j].attrvale) exit then
12. a=getposition(AAR[j].attrvale);
13. AAR.policycode.setposition(a)=1;
14. end if
15. end for
16. Step 2: Retrieve policy table.
17. for (i=0;i<Group.size;i++)  do    /*进行策略检索*/
18.    if   (Group.[i].groupbinary&AAR.groupbinary)==Group[i].groupbinary) then
19. for(g=0;g<Group[i].size;g++)  do
20. if (Group[i].ruleset[g]. policycode&AAR.policy_code == (Group[i].ruleset[g].policycode)
21. then
22. decide();     /*作出决策*/
23. end if
24. end if
25. end for
```

根据前述的对 AAR 进行二进制标识和编码的算法来看：本文对 AAR 处理的最多的次数是 ARR 的数量，所以其时间复杂度位 $O(n)$。当进行策略检索时，要对每个分组的每条规则进行判断，所以其时间复杂度位 $O(n^2)$，通常，n 取值不大，如后面实验中，本文取 n=6。

3　实验结果与分析

为评估基于二进制序列策略检索方法的效率，本文在 Win10 平台上利用 C++语言在 Qt Creator 中编写测试代码，并用 Matlab 作为数据分析工具。通常，评估检索方法需要大量的 ABAC 策略和 AAR，不幸的是，由于访问控制策略属于机密性内容，无法获得真实的行业数据。通常，路由器上的访问控制策略条数是 300-500 条。如文献[13] 选取策略总数量为 105、210、315、433、525 的五种情况进行实验。本文中，为了说明算法的高效性以及出现大规模的访问控制策略问题，选取更多的访问控制策略条数，如：2000、2500 和 3000。因此，参考了上述提到的属性信息，实现一个简单的策略和访问控制请求生成器，产生了 4000 条 ABAC 策略和 2000 条 AAR，其中每条策略包含一个访问控制规则。为了保证二进制标识的有效性，参考了文献[17]的数据分类方法进行标准化。本文的实验以这些数据为基础从策略预处理、策略评估时间、策略检索总时间三个方面测试本文提出的检索方法的效率。同时，为保证测试结果的一般性，以下实验得到的数据都是进行 10 次实验并对其求平均值的结果。实验环境为：CPU: 11th Gen Intel(R)Core(TM)i5-1135G7@2.40GHZ 2.42GHz, RAM:16.00GB; Qt Creator 3.3.0(opencesource) Based on Qt 5.4.0(MSVC 2010,32 bit)，Matlab 版本：8.6.0.267246 (R2015b)。

在下列实验中，考虑到本文提出的二进制标识和编码方法与文献[11]和[13]方法都属于分组型和前缀型查找方法，本文用 SG-PRM 表示文献[11]提出的基于前缀的策略检索方法，AG-PRM 表示文献[13]提出的基于属性分组的访问控制策略检索方法，而 B-S-PRM 指本文提出的策略检索方法。实验对比都是基于相同的访问控制策略和访问控制请求。策略复杂度指的是策略中每条规则中所包含的属性键值对的个数。

3.1　策略预处理时间

策略预处理是在访问控制的准备阶段完成的，主要对策略属性进行部署。本文提出的策略检索方法的策略预处理时间指的是 PAP 对策略集合中的每一条规则添加二进制标识、分组、编码的时间。SG-PRM 在策略预处理阶段对策略进行二进制标识。AG-PRM 在策略预处理阶段对策略进行二进制标识，分组。由图 2 可知，当把策略数目设置为 500，1000，1500，2000，2500 时，B-S-PRM, SG-PRM 和 AG-PRM 策略检索时间都随着策略数目的增多检索时间也逐渐增长，且呈线性增长趋势。其中，AG-PRM 和 SG-PRM 这两种策略检索方法的策略预处理时间比较接近，且低于 B-S-PRM 策略检索方法的策略预处理时间，这是因为本文提出的策略检索方

法在前两种策略检索方法的基础上增加了二进制编码的过程。基于二进制序列的策略检索方法(B-S-PRM)对策略的分析相比其他两种策略检索方法确实更费时间。但是，对基于属性访问控制策略的预处理发生在访问控制系统的准备阶段，并不会对策略检索时间造成任何影响。因此，本文的策略检索方法并不影响用户的实时体验。

3.2　策略评估时间

策略评估是指 PDP 在执行阶段完成的。主要指 PDP 检索符合访问控制请求的规则并作出决策的过程。图 3(a)(b)和(c)显示了不同策略规则数目条件下，增加访问控制请求数量，观察策略评估时间的变化。当策略数目分别设置位为 2000，2500，3000，这三种策略检索方法的检索时间都随着访问控制请求的增多而加长。其中，SG-PRM 在策略检索阶段访问控制请求的前缀和策略的前缀做逻辑或运算，运算结果与策略的前缀一致再匹配属性信息是否相符。AG-PRM 对策略集进行基于属性的分组而缩减策略检索的范围。

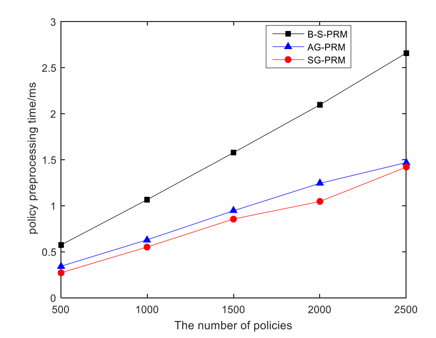
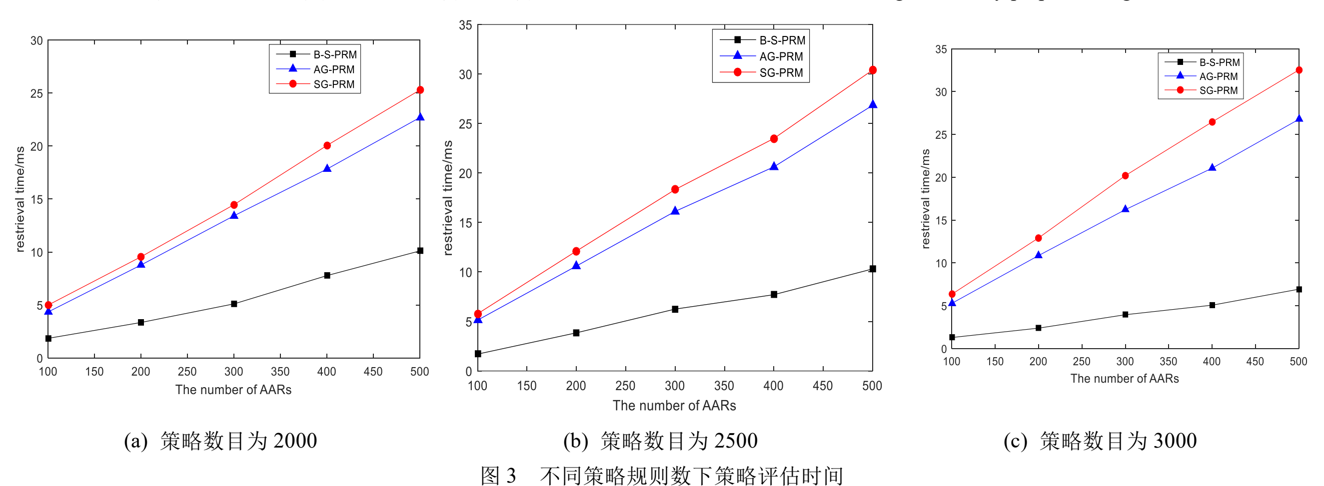


图 2　策略预处理

Fig. 2　Policy preprocessing



(a) 策略数目为 2000　　(b) 策略数目为 2500　　(c) 策略数目为 3000

图 3　不同策略规则数下策略评估时间

Fig. 3　The policy evaluation time under different policy rules

通过图 3(a)(b)(c)可以看出，在不同的访问控制策略条件下，随着访问控制请求数量的增加，这三种策略检索方法的策略评估时间都逐渐加长。其中，本文提出的策略检索方法(B-S-PRM)的策略评估时间随着策略数量的增多呈下降趋势。当策略数量为 3000 时，B-S-PRM 的策略的评估时间约为策略数量为 2000 时的一半，其策略评估时间的波动范围也逐渐缩小,而其他两种策略检索方法的策略评估时间却没发生明显变化。SG-PRM 的策略评估时间在 4-32ms 之间波动，AG-PRM 的策略评估时间在 4-27ms 之间波动，B-S-PRM 在 1-11 之间波动，由此可以看出本文提出的策略检索方法的检索时间更加稳定,B-S-PRM 的策略检索效率约是 SG-PRM 和 AG-PRM 的 3 倍。这说明本文提出的策略检索方法能够应用于高策略的环境下。

3.3　策略检索总时间

基于属性访问控制策略的检索总时间包括 PEP 对 AAR 处理的时间，以及 PDP 对策略进行评估的时间。如图 4 所示，策略总数为 3000，随着访问控制请求数量的增加，三种策略检索方法对访问控制请求的处理时间都逐渐增长，且接近于线性增长。其中，SG-PRM 和 AG-PRM 对访问控制请求的处理时间比较接近，本文提出的策略检索方法对访问控制请求的处理时间约是 SG-PRM 和 AG-PRM 的两倍。这是因为在对访问控制的处理过程中，本文提出的策略检索方法更为复杂，SG-PRM 对访问控制请求添加前缀，B-S-PRM 需要对策略进行二进制编码和标识，AG-PRM 只需要到对访问控制请求添加分组标识和前缀标识。但是，通过图 5 可以看出其本文提出的策略检索方法虽然在 PEP 对 AAR 的处理花费了较长的时间,但其总的检索时间反而有大幅度下降。其中，B-S-PRM 的检索效率约比 SG-PRM 提高了 4 倍，比 AG-PRM 提高了 3.5 倍以上。随着访问控制请求数量增加，本文提出的策略检索方法更具有优势，策略检索时间更加稳定。因为，当策略增多时，不同于另外两种策略检索方法，本文提出的策略检索方法没有对规则内的属性值对进行遍历，而是选择合适的分组对本组内的规则遍历，从而减少了策略检索的范围。因此，检索效率得到极大的提高。

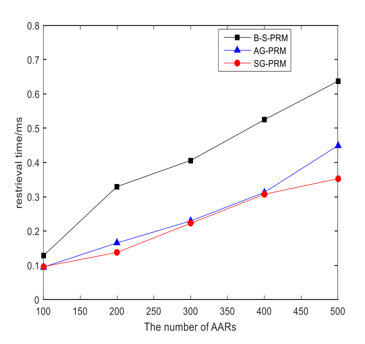


图 4　访问控制请求的处理时间

Fig. 4　The processing time of access control request

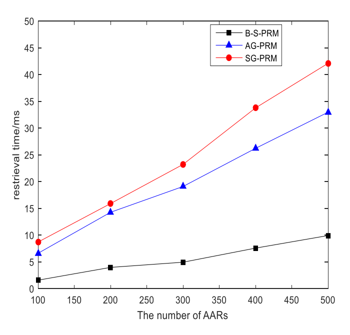


图 5　总的策略检索时间

Fig. 5　The total policy retrieval time

综上所述，本文提出的策略检索方法虽然在策略的预处理时间比其他两种策略检索方法时间更长，但是其策略的检索时间却显著的缩短了，检索效率优势明显。随着策略规模的增大，本文提出的策略检索方法的检索时间并未发生显著变化。应用到云计算、的基于属性访问控制的部署中，当用户的访问控制请求到来时，该方法能够缩短用户的等待时间，实时响应用户的访问控制请求。

4 结束语

随着云计算、物联网技术的兴起，当今的网络环境越来越趋于海量性和动态性，用户和资源规模巨大，主客体的访问环境也实时动态变化。基于属性访问控制作为一种保护客体资源的有效技术而被广泛应用。用户希望在较短的时间内得到访问客体资源的权限，从而缩短策略检索时间，减少对访问控制请求的响应时间。然而，现有的基于属性访问控制策略的检索方法应用于大规模访问的情况下存在着一定的不足。针对该问题，本文提出一种基于二进制序列的属性访问控制策略检索方法。该方法对基于属性访问控制请求和基于属性访问控制进行二进制标识和编码，并且根据二进制标识对策略进行分组。在进行检索时，首先，访问控制请求的二进制标识和组二进制标识进行逻辑与运算来选择符合要求的分组，过滤掉大量无关的策略。然后，访问控制请求的二进制编码与组内规则的二进制编码做逻辑运算来选择合适规则，减少了策略集合内规则的属性与访问控制请求属性匹配的过程。实验结果验证了本文提出的策略检索方法具有更低的策略检索时间，更高的检索效率。

在保证策略检索效率的情况下，考虑将本文的策略检索方法和属性加密相结合来进一步提高属性的安全性是未来的研究方向。另外，本文的提出策略检索方法只在理论层面，如把该策略与实际情况相结合进行优化，以适应于未来实际的应用也是将来的研究方向。

参考文献：

[1] PanJun Sun. Security and privacy protection in Cloud Computing: discussions and challenges [J]. Journal of Network and Computer Applications. 2020, 160 (102642): 1-22

[2] Servos D, Osborn S L. Current research and open problems in attribute-based access control [J]. ACM Computing Surveys, 2017, 49 (4): 1-45.

[3] OASIS XACML. eXtensible access control Markup language XACML version 3. 0 [S]. OASIS Standard, Munich, 2011.

[4] D. Nabil, H. Slimani and H. Nacer, *et al.* ABAC conceptual graph model for composite Web services [C]// The 5th IEEE International Congress on Information Science and Technology, Marrakech, Morocco, IEEE 2018: 36-41

[5] 罗杨，沈晴霓，吴中海. 一种新的访问控制策略描述语言及其权限划分方法 [J]. 计算机学报, 2018, 41 (6): 969-986 (Luo Yang, Shen Qingni, Wu Zhonghai. A novel access control policy specification language and its permission classification method [J]. Chinese Journal of Computer, 2018, 41 (6): 969-986)

[6] Jun Ho Huh, Rakesh B. Bobba, Tom Markham, et. al. Next-Generation access control for distributed control systems. IEEE Internet computing, 2016, 20 (5): 28-37

[7] Working DRAFT Information technology-Next Generation Access Control –Generic Operations and Data Structures (NGAC-GOADS) , INCITS 499-2013, American National Standard for Information Technology, American National Standards Institute, 2014.

[8] Mohamed Mejri, Hamdi Yahyaoui, Azzam Mouradc, *et al.* A rewriting system for the assessment of XACML policies relationship [J]. Computers & Security, 97 (2020) , 2020, 1-12.

[9] Santiago Pina Ros, Mario Lischka, Félix Gómez Mármol. Graph-based XACML evaluation [C]// In Proceedings of 17th ACM symposium on Access Control Models and Technologies (SACMT) , New York, NY, 2012, 83-92.

[10] D. H. Niu, J. F. Ma, Z. Ma, C. N. Li, L. Wang. HPEngine: High performance XACML policy evaluation engine based on statistical analysis [J]. Journal of Communication. 2014, 35 (8): 206–215.

[11] 邹佳顺，张永胜. ABAC 中基于前缀标记运算的策略检索方法 [J]. 计算机工程与设计, 2015, 36 (11): 2943-2947. (Zou Jiashun, Zhang Yongsheng. Policy retrieval method based on prefix sign operation in ABAC model [J]. Computer Engineering and Design, 2015, 36 (11): 2943-2947)

[12] Liu M P, Cheng Y, Hao Li, *et al.* An efficient attribute-based access control (ABAC) policy retrieval method based on attribute and value levels in multimedia networks [J] , Sensors, 2020, 20 (6) , 1-15.

[13] 黄美蓉，欧博. 基于属性分组的访问控制策略检索方法 [J]. 计算机应用研究, 2020, 37 (10): 3096-3100+3106. (Huang Meirong, Ou Bo. Attribute and RBAC based hybrid access control model [J]. Application Research of Computers, 2020, 37 (10): 3096-3100+3106)

[14] Vincent Hu, David F. Ferraiolo, D. Richard Kuhn, *et al.* Implementing and managing policy rules in attribute based access control [C]// 2015 IEEE 16th International Conference on Information Reuse and Integration, San Francisco, California, IEEE Press, 2015, 518-525

[15] Ronit Nath, Saptarshi Das, Shamik Sural. PolTree: A data structure for making efficient access decisions in ABAC [C]// In The 24th ACM Symposium on Access Control Models and Technologies (SACMAT'19) , Toronto, ON, ACM Press, 2019, 25-35

[16] Zhao Z. Y. , Sun L. Attribute-based access control with dynamic trust in a hybrid cloud computing environment [C]// Proceedings of the 2017 International Conference on Cryptography, Security and Privacy. New York, ACM Press, 2017: 112-118

[17] Shaikh R A, Adi K, Logrippo L. A data classification method for inconsistency and incompleteness detection in access control policy sets [J]. International Journal of Information Security, 2017, 16 (1): 91-113